LUENTO 1

Käyttöjärjestelmät II

Kertaus: KJ-I ja RIO (KJ2'n osalta)

Ch 1-8, 11-12 [Stal05]
Kustakin luvusta enemmän tai vähemmän alkuosa

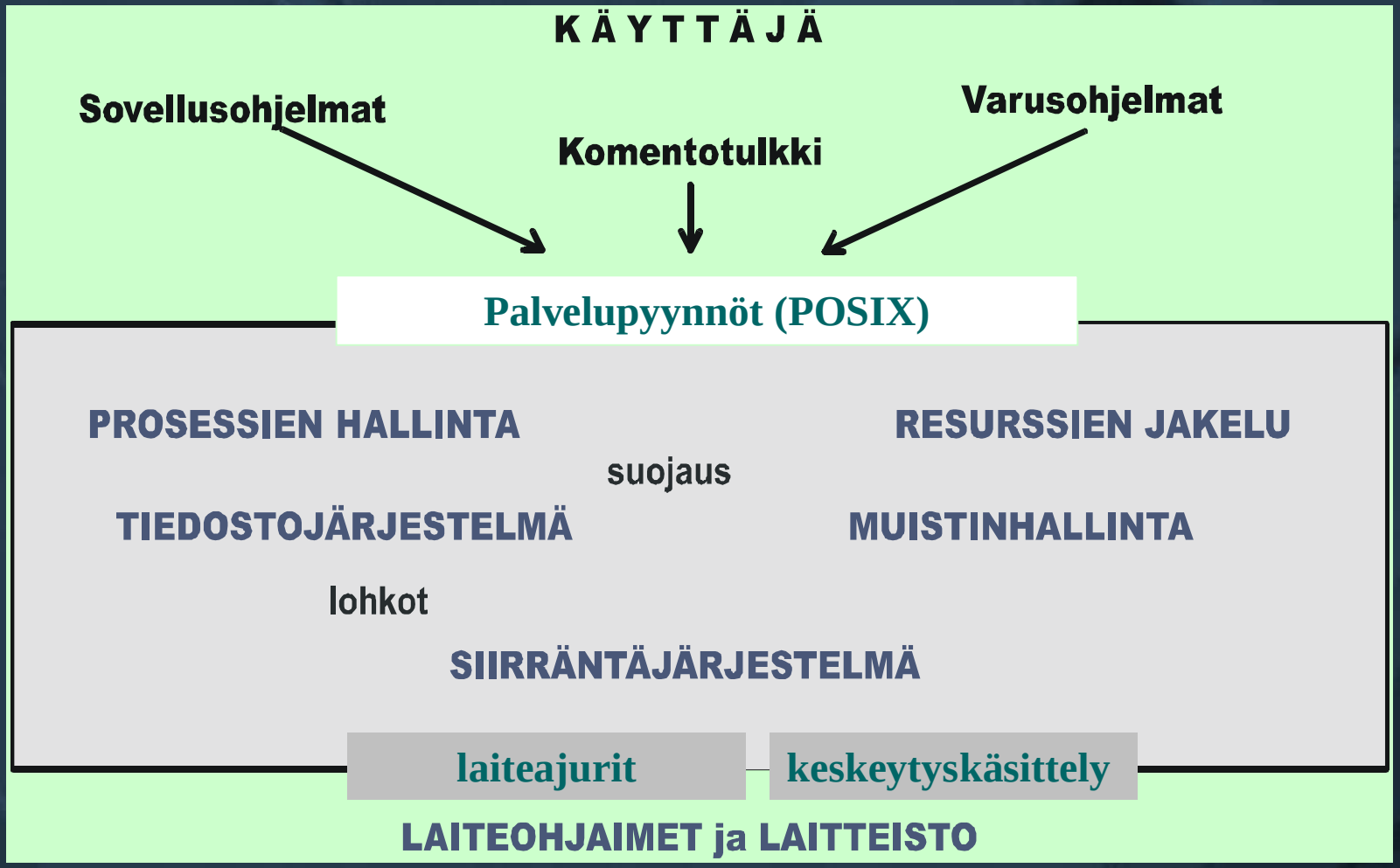
Jos jokin asiat tässä tuntuvat hatarilta, niin
(a) kysykää ja/tai (b) kerratkaa kirjasta

Jatkossa nämä asiat otaksutaan hyvin osatuksi.

Yleiskuva

- **Yleiskatsaus** (Ch 1 [Stal05])
  - Käskysykli, keskeytys, muistihierarkia, välimuisti, siirrännän tekniikat
- **Yleistä käyttöjärjestelmästä** (Ch 2.1-2.4 [Stal05])
  - KJ:n tehtäviä ja toimintoja, KJ:n evoluutiotarina, keskeiset KJ:n osat, nykyaikaisen KJ:n piirteet

Tietokonejärjestelmä

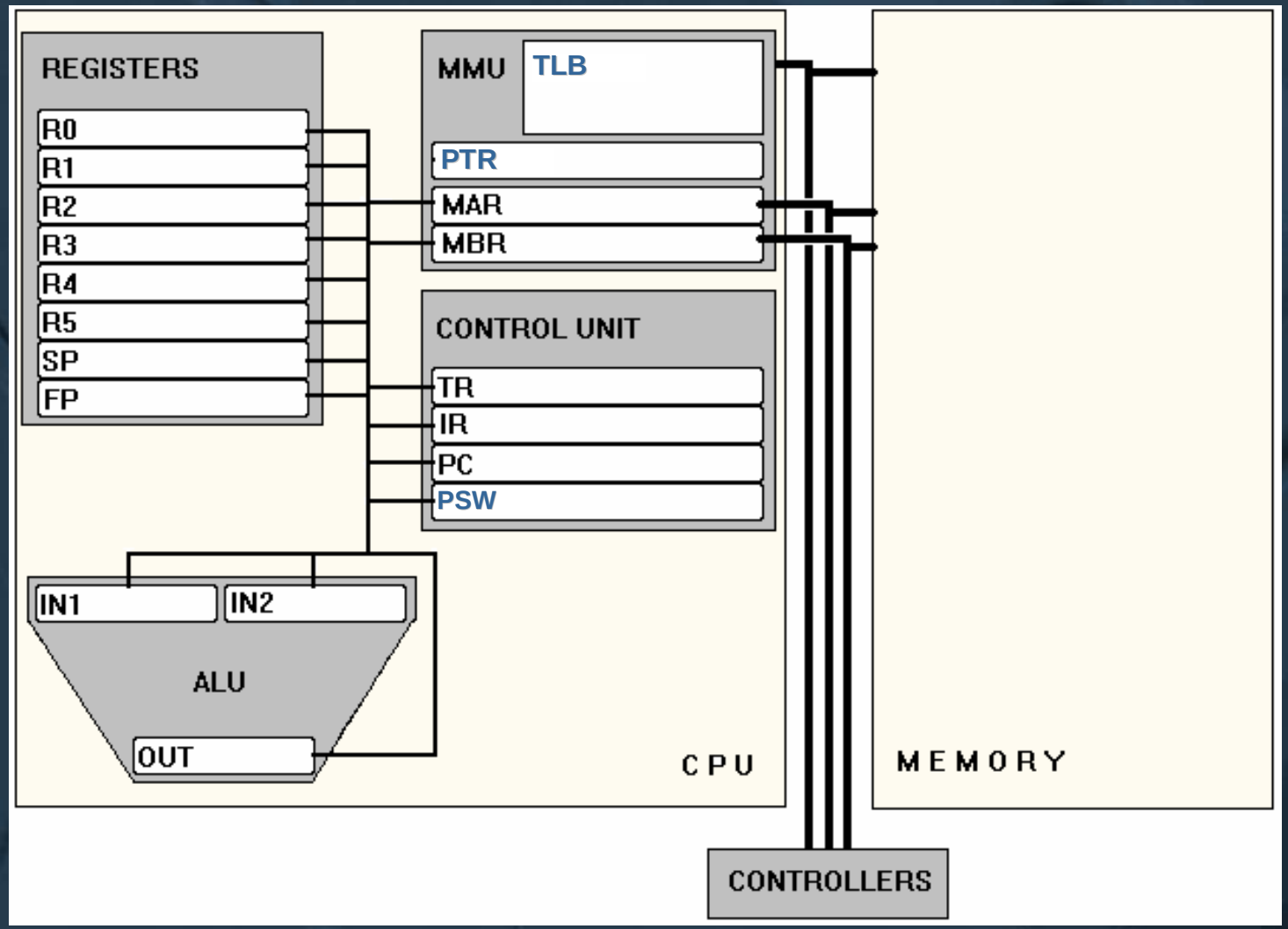


Käyttöjärjestelmät II

KJ ja laitteistopiirteet

Stal05 Ch 1

Keskusyksikkö (Kuva 3.1, Häkk98)



KJ ja laitteistopiirteitä

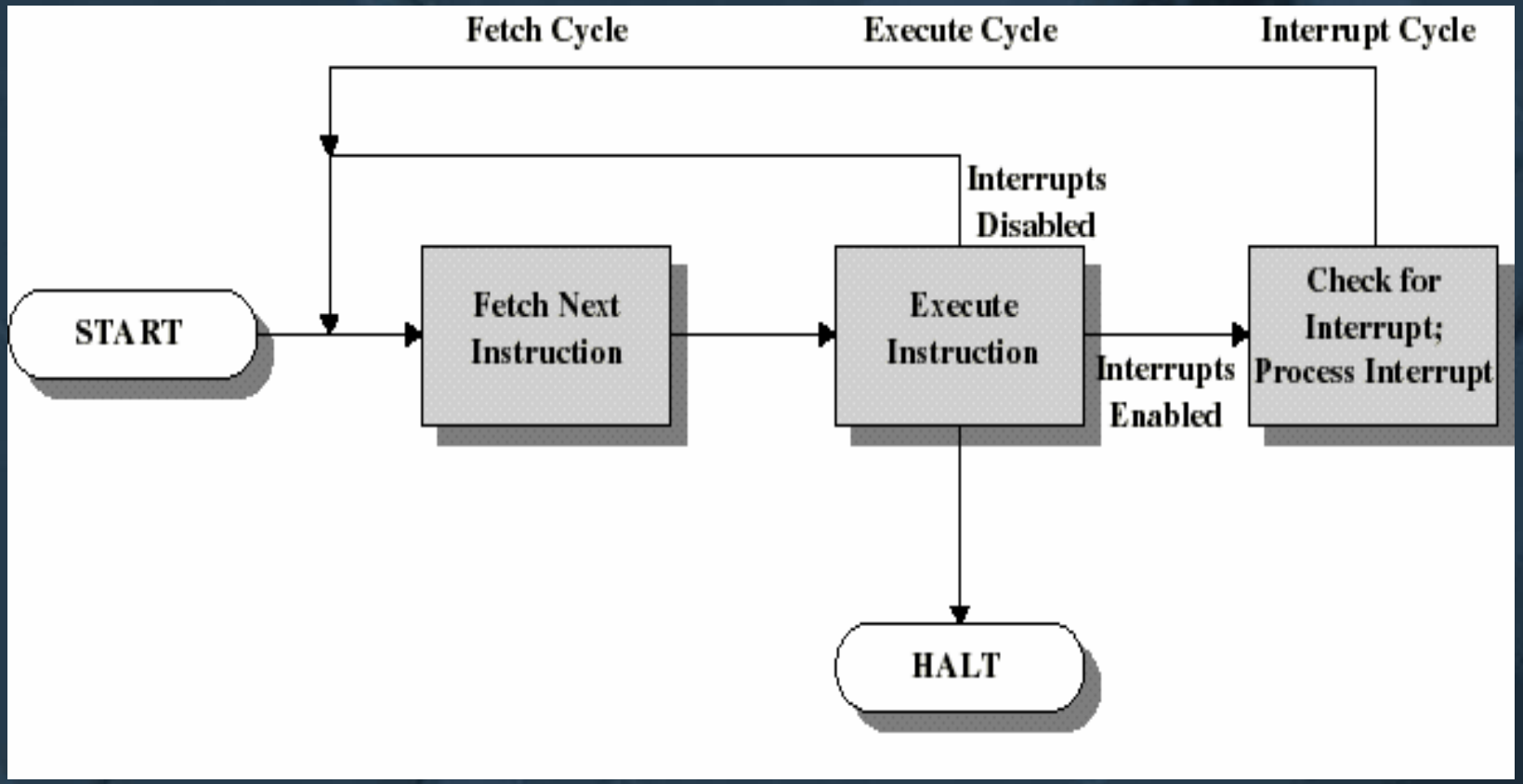
- **Etuoikeutetut käskyt** (Privileged Instructions)
  - siirräntäkäskyt (fyysinen siirräntä)
  - MMU käytön vaatimat asetukset
  - keskeytysten esto ja salliminen
  - jos sovellus yrittää käyttää näitä käskyjä, tuloksena poikkeus ‘tuntematon käskykoodi’
- **Etuoikeutettu tila / käyttäjätila** (Supervisor/User mode)
  - bitti PSW:ssä
  - vain laitteisto voi asettaa etuoikeutetun tilan
  - CPU suorittaa etuoikeutetun käskyn vain, jos on etuoikeutetussa tilassa
- **Jakamaton test-and-set käsky (tai muu vastaava)**
  - poissulkeminen, synkronointi

KJ ja laitteistopiirteitä (jatk)

- **Osoitemuunnos ja muistinsuojaus**
  - KJ suojattava sovellukselta, sovellukset toisiltaan
  - CPU:n tehtävä ajonaikainen osoitemuunnos ja tarkistettava muistiosoitteet
    - laitteistoon tarvitaan MMU
    - MMU:ssa osoitemuunnospuskuri TLB
- **Keskeytysmekanismi**
  - hallittu kontrollin siirto KJ:n ja sovelluksen välillä
  - bitti PSW:ssä, keskeytyskäsittelyn alku laitteistotoiminto
- **Kellokeskeytys**
  - ettei yksi sovellus valloita koko laitteistoa
  - viimeistään kello aiheuttaa keskeytyksen
    - kontrolli taas KJ:lle

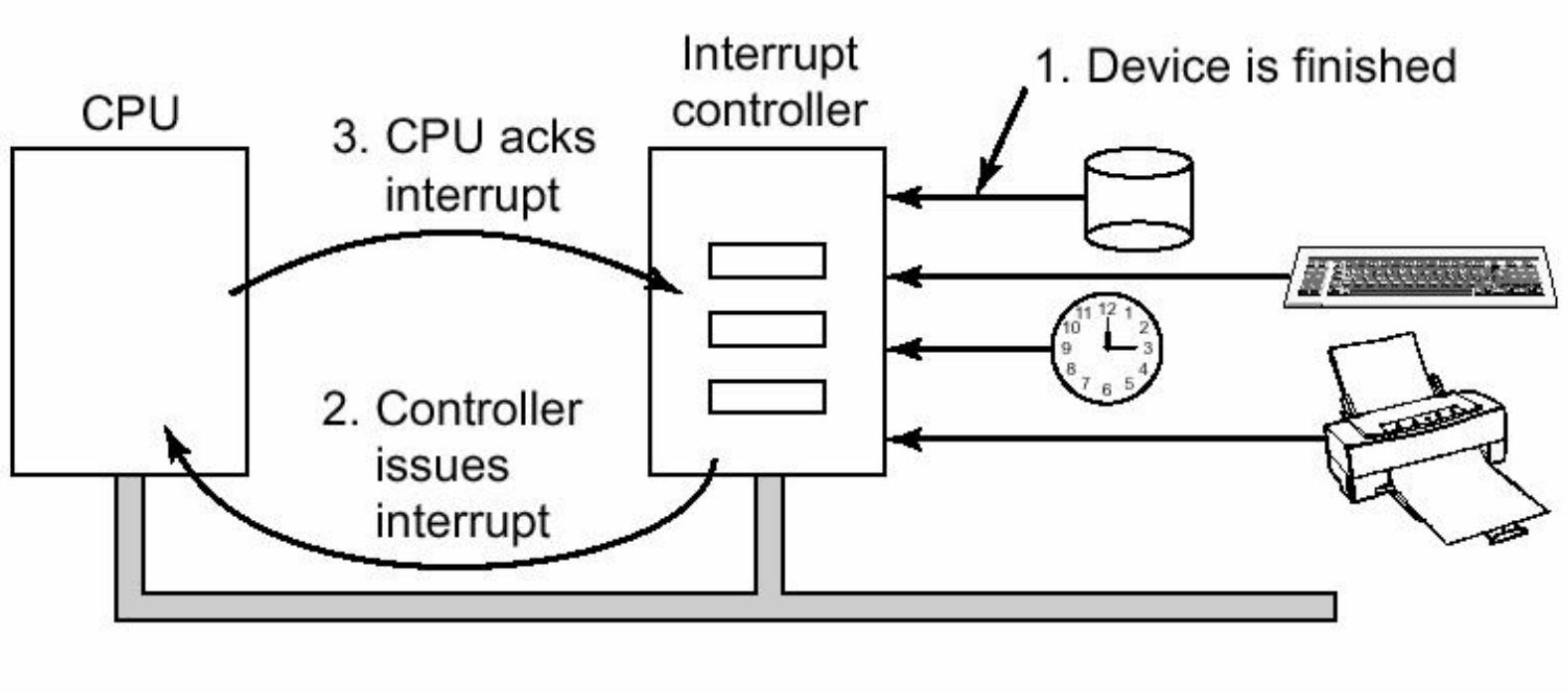
Käskysykli

Fig 1.7 [Stal05]



Keskeytys

Fig. 5-5 [Tan01]



Keskeytyskäsittely (kesk. käsittelijä)

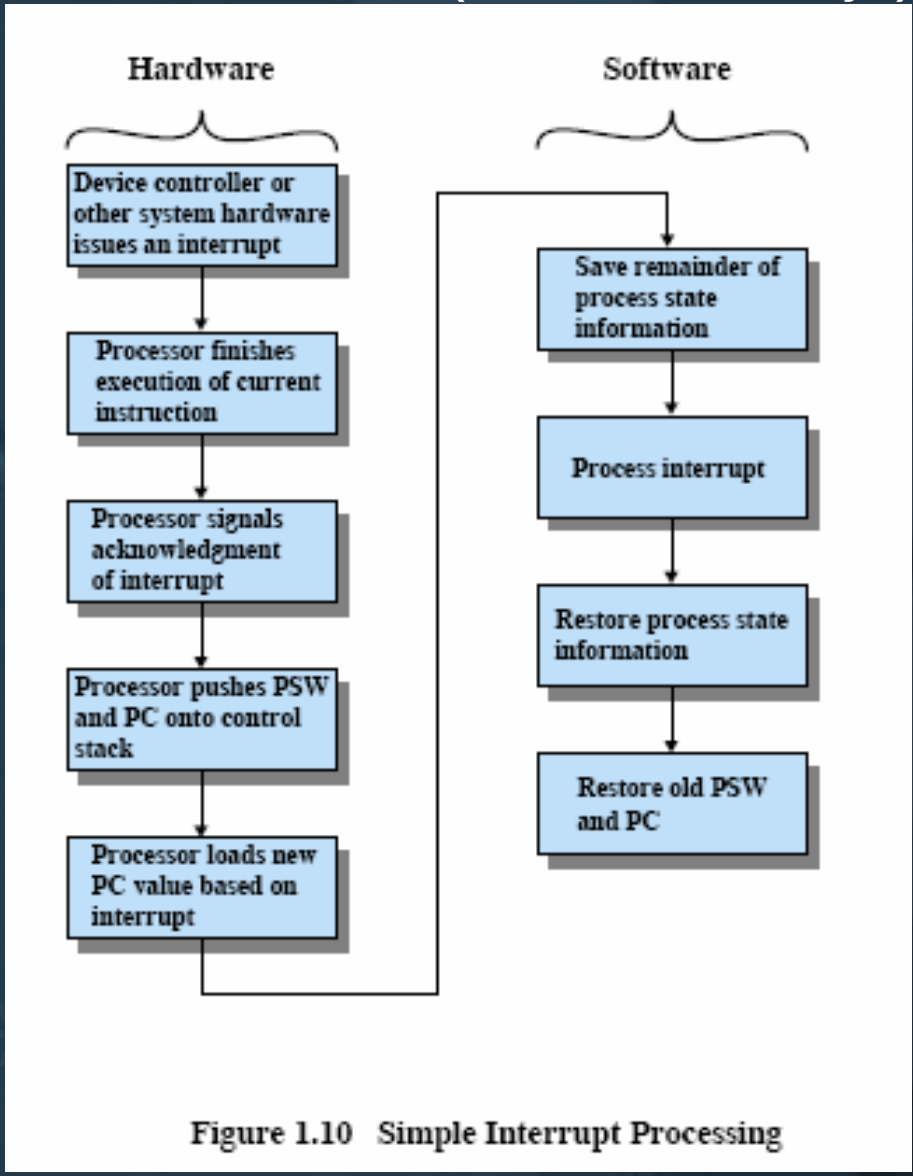


Figure 1.10 Simple Interrupt Processing

Etuoikeutettu tila vs. käyttäjätila

Keskeytysten esto vs. salliminen

s. 21-26 [Stal05]

Käyttöjärjestelmät II

KJ ja palvelupyynnöt

Ch 2.1-4 [Stal05]

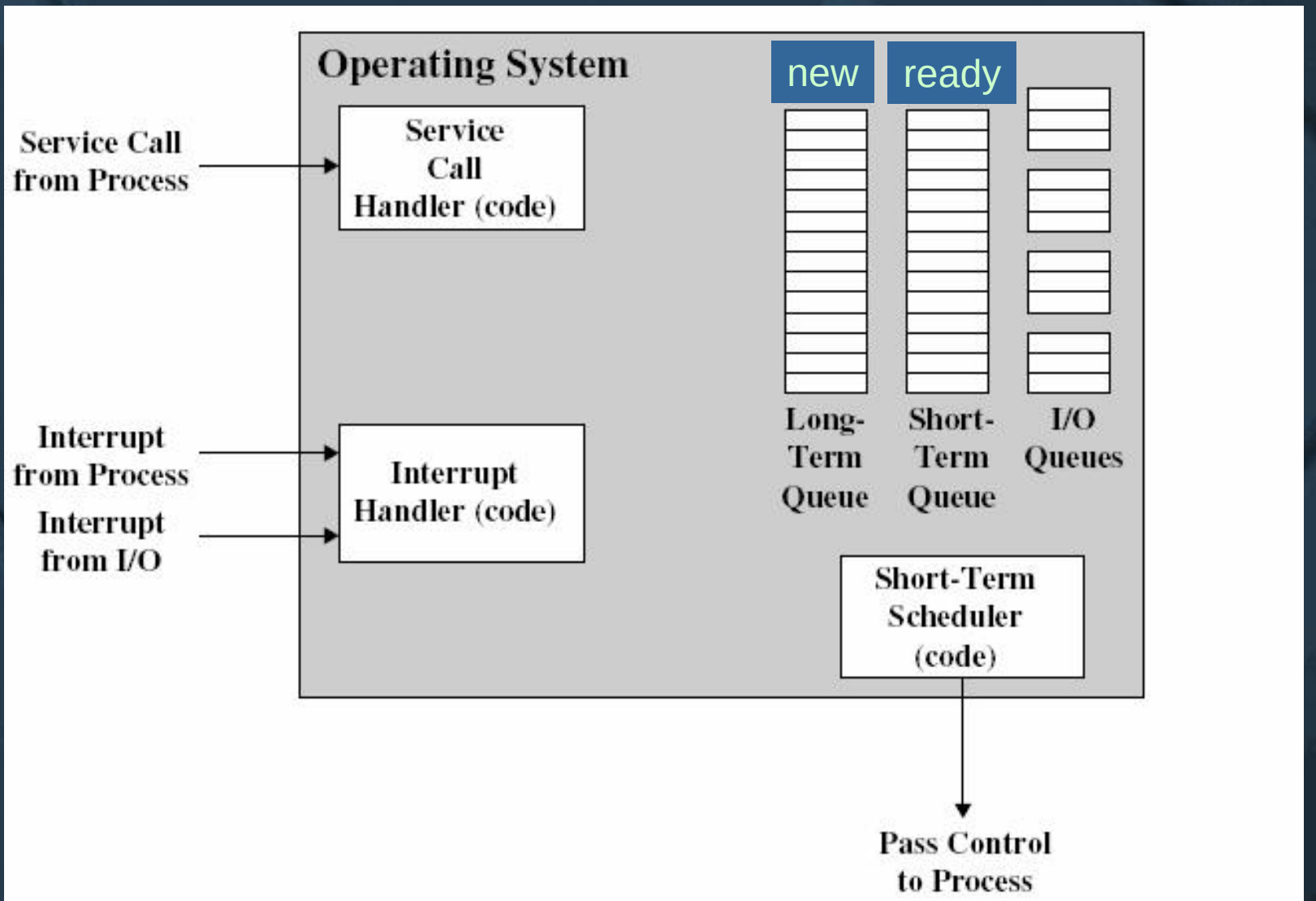
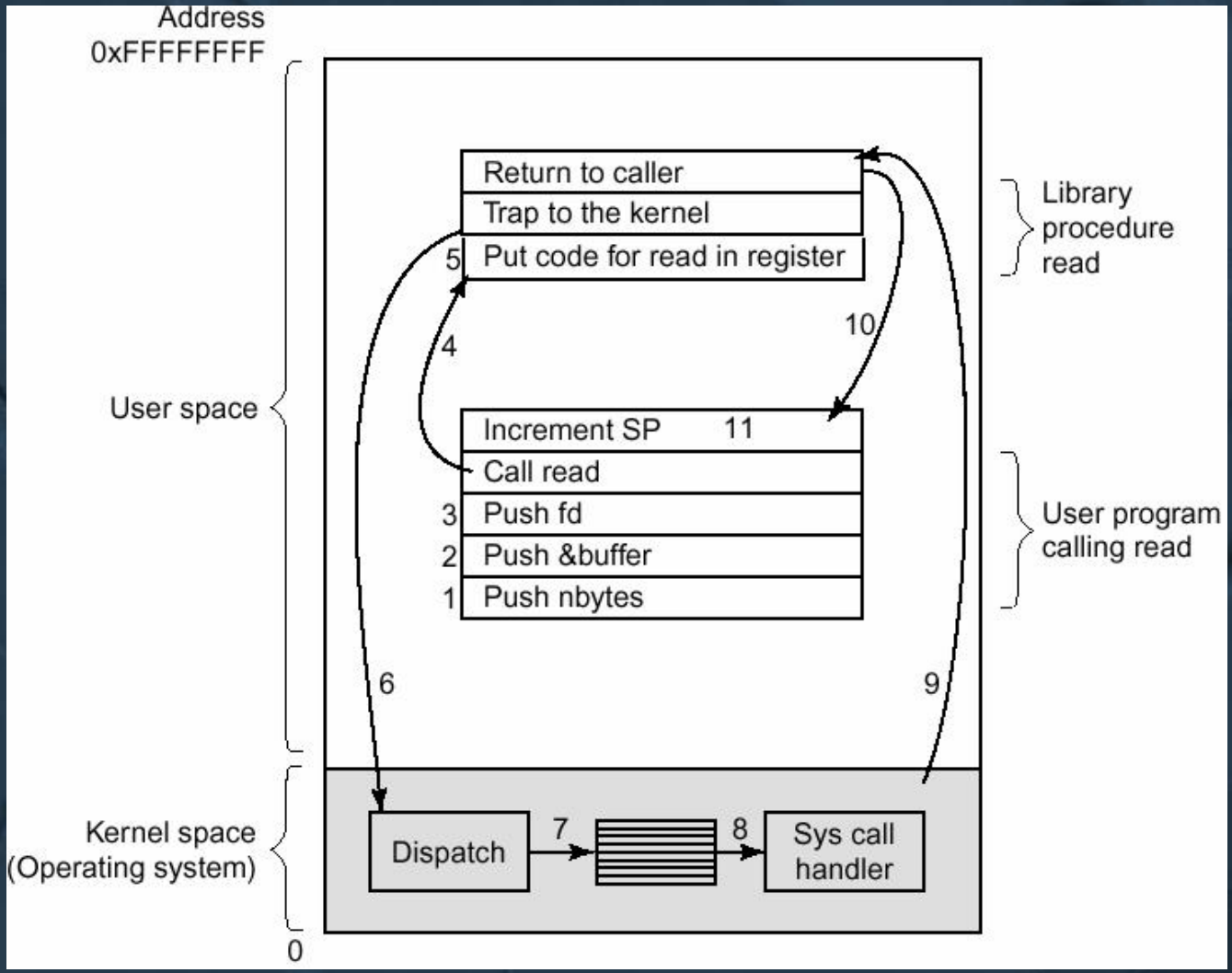


Figure 2.11 Key Elements of an Operating System for Multiprogramming

read(fd, buffer, nbytes) Fig 1-17 [Tan01]



POSIX palvelupyyntöjä

Fig 1-18 [Tan01]

Process management

Call	Description
pid = fork( )	Create a child process identical to the parent
pid = waitpid(pid, &statloc, options)	Wait for a child to terminate
s = execve(name, argv, environp)	Replace a process' core image
exit(status)	Terminate process execution and return status

File management

Call	Description
fd = open(file, how, ...)	Open a file for reading, writing or both
s = close(fd)	Close an open file
n = read(fd, buffer, nbytes)	Read data from a file into a buffer
n = write(fd, buffer, nbytes)	Write data from a buffer into a file
position = lseek(fd, offset, whence)	Move the file pointer
s = stat(name, &buf)	Get a file's status information

jne, kaikkiin KJ:n perustehtäviin liittyen... **ks. man syscalls**

POSIX: http://www.unix-systems.org/single_unix_specification/

Käyttöesimerkki

Fig 1-19 [Tan01]

”Riisuttu” komentotulkki

```
#define TRUE 1

while (TRUE) {                                  /* repeat forever */
      type_prompt( );                           /* display prompt on screen */
      read_command(command, parameters);        /* read input from terminal */

      if (fork( ) != 0) {                       /* fork off child process */
            /* Parent code. */
            waitpid(–1, &status, 0);            /* wait for child to exit */
      } else {
            /* Child code. */
            execve(command, parameters, 0);     /* execute command */
      }
}
```

Huom: *fork()* palauttaa 0 ”lapselle” ja lapsen id’n ”isälle”

WIN32 vs. UNIX API Fig. 1-23 [Tan01]

UNIX	Win32	Description
fork	CreateProcess	Create a new process
waitpid	WaitForSingleObject	Can wait for a process to exit
execve	(none)	CreateProcess = fork + execve
exit	ExitProcess	Terminate execution
open	CreateFile	Create a file or open an existing file
close	CloseHandle	Close a file
read	ReadFile	Read data from a file
write	WriteFile	Write data to a file
lseek	SetFilePointer	Move the file pointer
stat	GetFileAttributesEx	Get various file attributes
mkdir	CreateDirectory	Create a new directory
rmdir	RemoveDirectory	Remove an empty directory
link	(none)	Win32 does not support links
unlink	DeleteFile	Destroy an existing file
mount	(none)	Win32 does not support mount
umount	(none)	Win32 does not support mount
chdir	SetCurrentDirectory	Change the current working directory
chmod	(none)	Win32 does not support security (although NT does)
kill	(none)	Win32 does not support signals
time	GetLocalTime	Get the current time

Käyttöjärjestelmät II

Prosessin hallinta

Ch 3.1-4, 4.1 [Stal05]

Prosessi

= Suoritettavaksi otettu ohjelma

koodi muistissa (voi olla yhteiskäytössä)

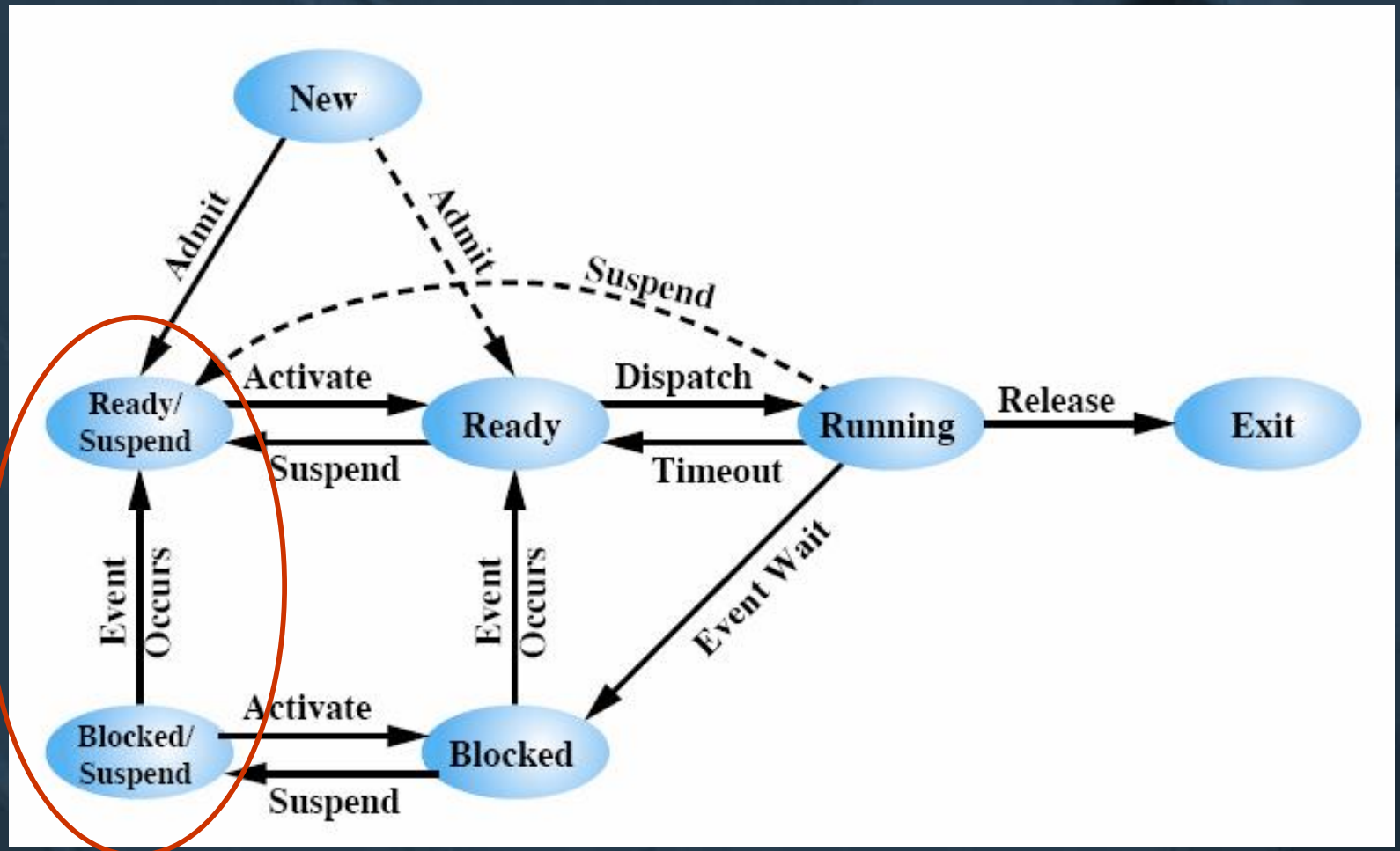
oma data-alue ja pino muistissa (muuttujat)

+ KJ:n ylläpitämät rakenteet, PCB

- CPU:n dataa prosessin suorittamisesta
  - prosessorin rekistereiden arvot
- KJ:n dataa prosessin hallitsemiseksi
  - tunnistus (pid, ppid), omistaja (uid, gid)
  - vuorottamisessa tarvittavaa tietoa
  - muistinhallinnassa tarv. tietoa (osoitin sivutauluun)
  - tiedostokuvaajataulu
  - …

(ks. Table 3.5 [Stal05])

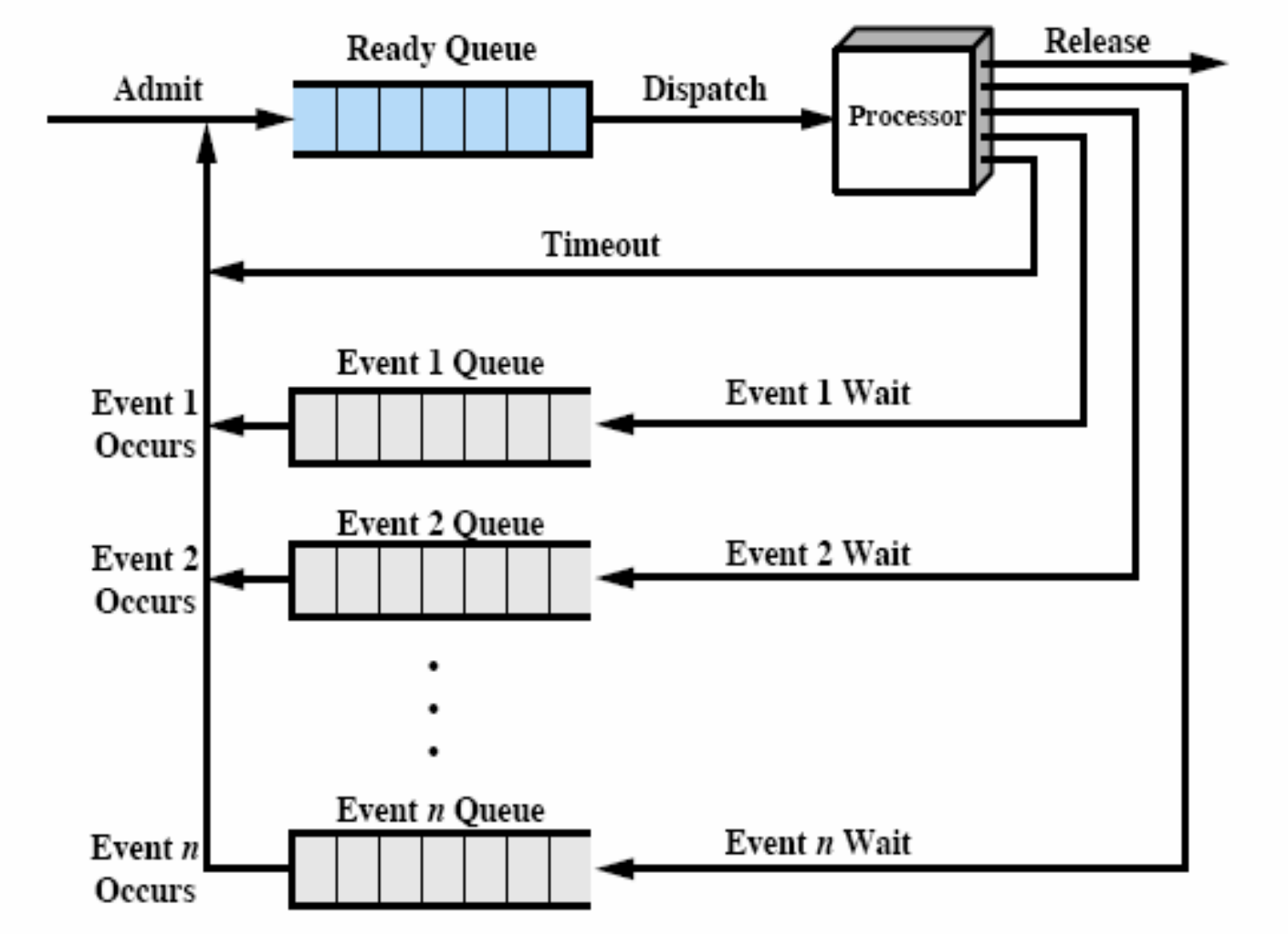
Prosessin tilakaavio Kuva 3.9 [Stal05]



”Levyllä, swapped out” ”Muistissa”

Prosessijonot

Kuva 3.8 [Stal05]

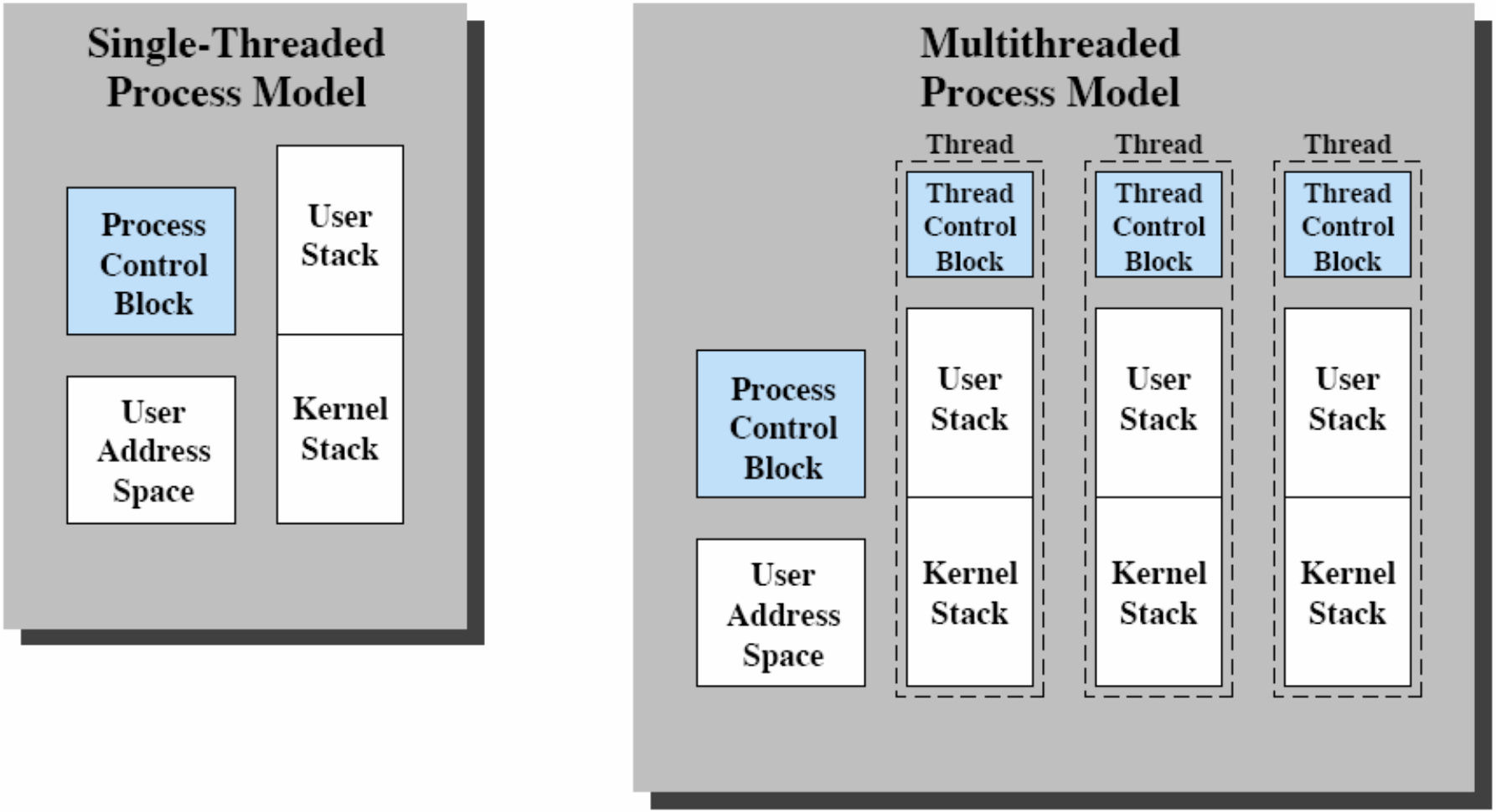


Milloin prosessinvaihto?

- **Vain keskeytyksen jälkeen**
  - ei kuitenkaan aina!
- **Kun CPU siirtynyt suorittamaan KJ:tä**
  - Palvelupyyntö
    - jonka seurauksena prosessi joutuu odottamaan
  - Poikkeus
    - prosessin suorituksessa virhe
    - prosessi joutuu exit-tilaan ja tapetaan
  - Ulkoinen keskeytys
    - siirräntä valmistuu
    - prosessin aikaviipale (50 ms - 100 ms) täynnä
- **Vuorottaja valitsee**
  - talleta edellisen prosessin tiedot CPU:sta sen PCB:hen
  - kopioi seuraavan prosessin tiedot PCB:stä CPU:hun

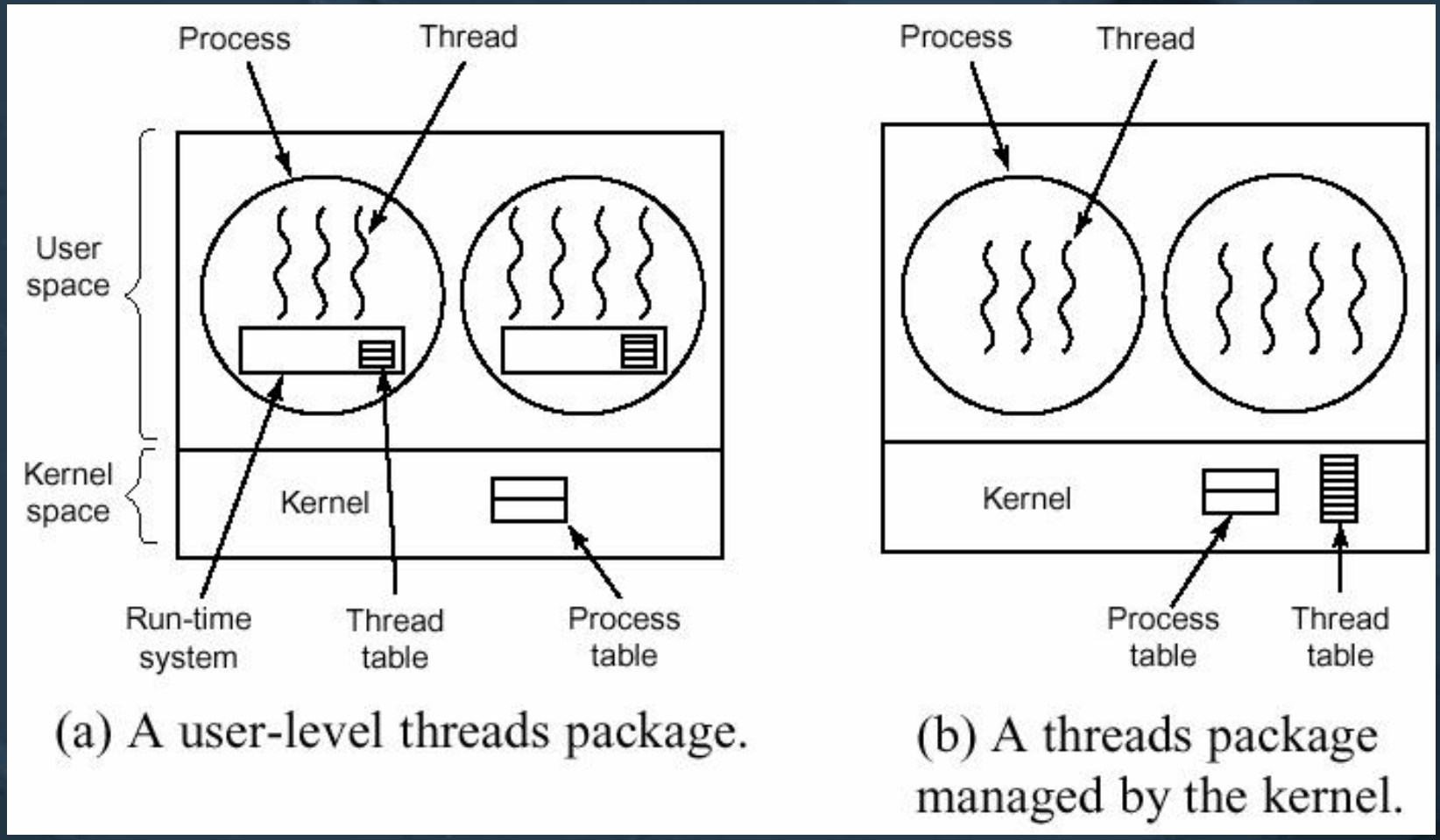
Säikeet

Fig 4.2 [Stal05]



Säikeet

Fig 2-13 [Tan01]



Käyttöjärjestelmät II

Samanaikaisuuden hallinta

Ch 5, 6.1-6 [Stal05]

Poissulkemisongelma

- Mutex määrittely
- Ohjelmistoratkaisu
- Laitetuki ongelman ratkaisuun
- Lukko vai semafori? Milloin?
- Monitorit ja niiden toteutus
- Lukijat ja kirjoittajat (synkronointi)

- Miten tiedät, että ratkaisusi on oikein?

Lukkiutumisongelma

- Lukkiutumisen havaitseminen
- Lukkiutumisen välttäminen
- Lukkiutumisen purkaminen
- Aterioivat filosofit

Käyttöjärjestelmät II

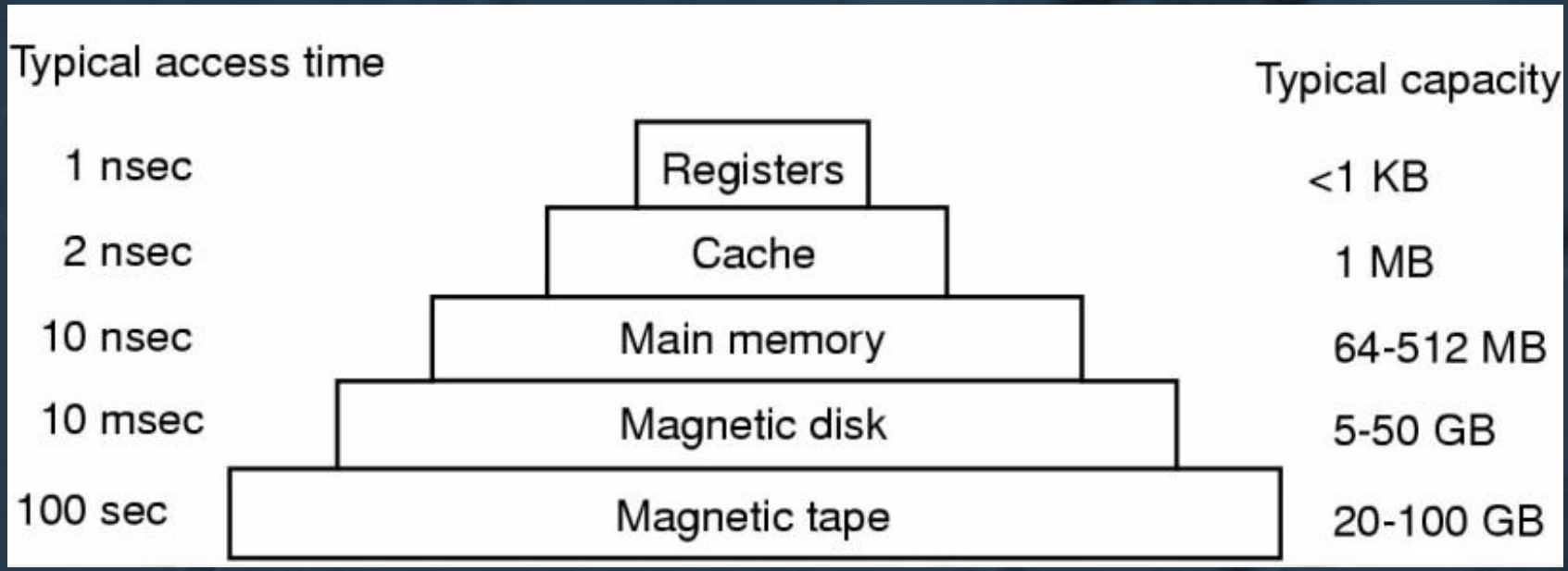
Muistinhallinta

Virtuaalimuisti

Ch 7, 8.1 [Stal05]

Muistihierarkia

Fig 1.14 [Tan01]

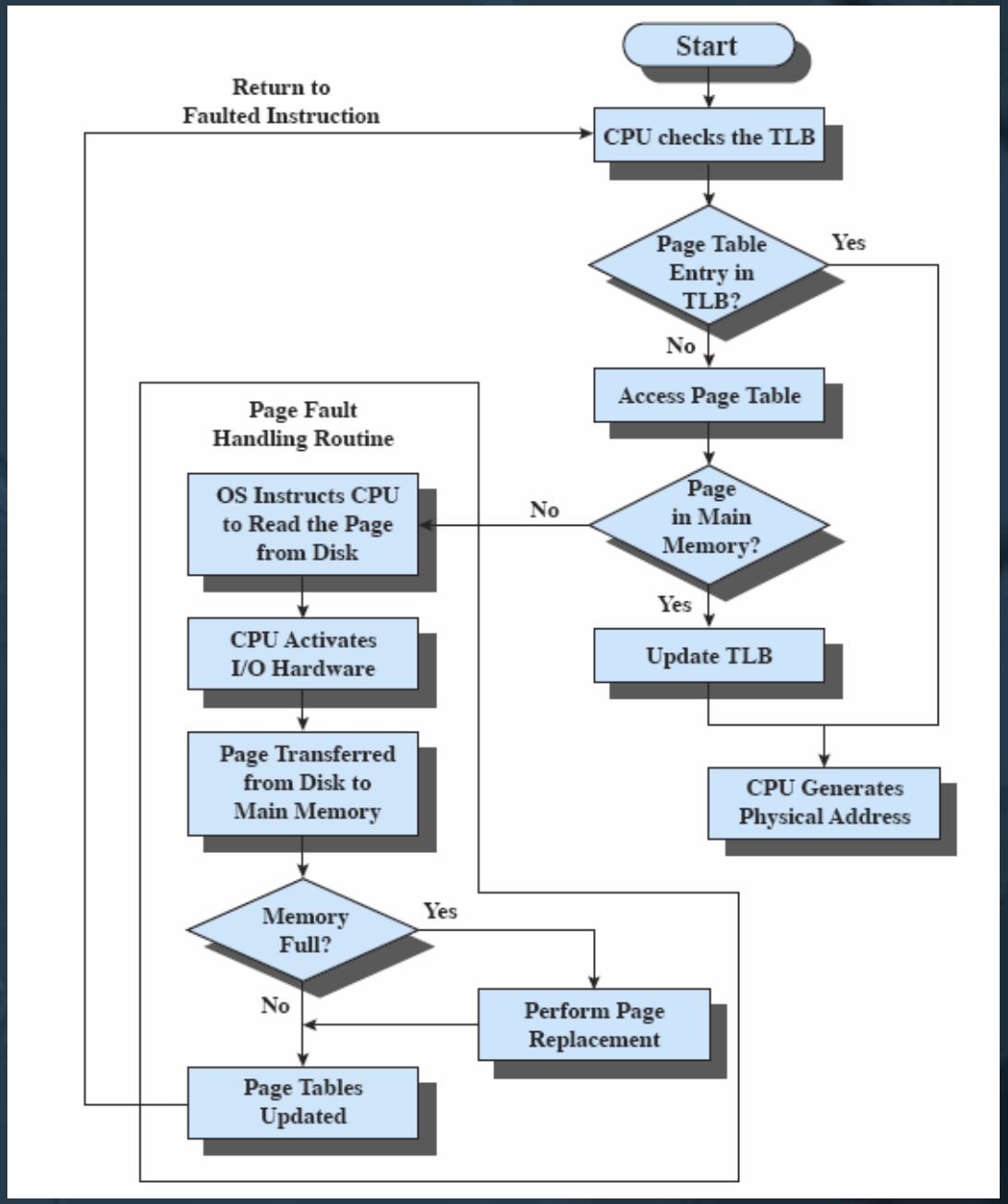


nano = 10^{-9}, mikro = 10^{-6}, milli = 10^{-3}

Virtuaalimuisti (Table 8.1 [Stal05])

Sivutus	Segmentointi
Keskusmuisti jaettu vakiokokoisiin sivutiloihin	
KJ jakaa prosessin vakiokokoisiin sivuihin	Ohjelmoija/kääntäjä jakaa prosessin vaihtelevankokoisiin segmentteihin
Prosessikoht. sivutaulut: missä sivutilassa sivu sijaitsee	Prosessikoht. segmenttitaulut: segmentin alkuos. ja pituus
Virt.os: (sivu, siirtymä)	Virt.os: (segmentti, siirtymä)
Sisäinen pirstoutuminen	Ulkoinen pirstoutuminen muistin tiivistämistarve
Vapaiden sivutilojen lista	Vapaiden muistialueiden lista
Kaikki sivut ei muistissa - läsnäolobitti sivut-alkiossa - poisto / korvauspolitiikka	Kaikki segmentit ei muistissa - läsnäolobitti Seg.t.-alkiossa - poisto / korvauspolitiikka

MMU ja osoitteen-muunnos



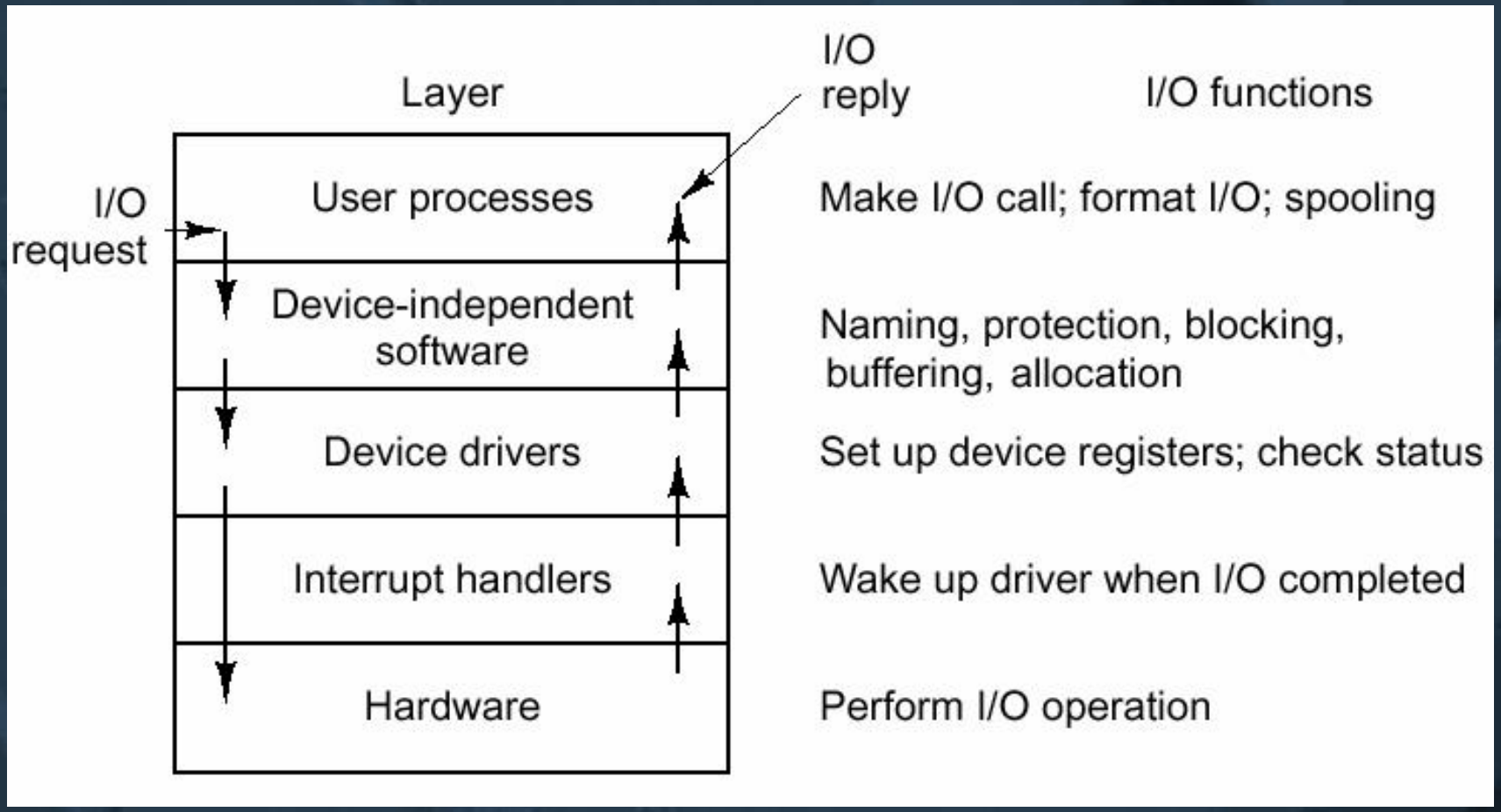
Kuva 8.8.[Stal05]

Käyttöjärjestelmät II

Siirräntä

Ch 11.1-4 [Stal05]

Siirrännän hierarkia Fig. 5-16 [Tan01]



Laiteajurit

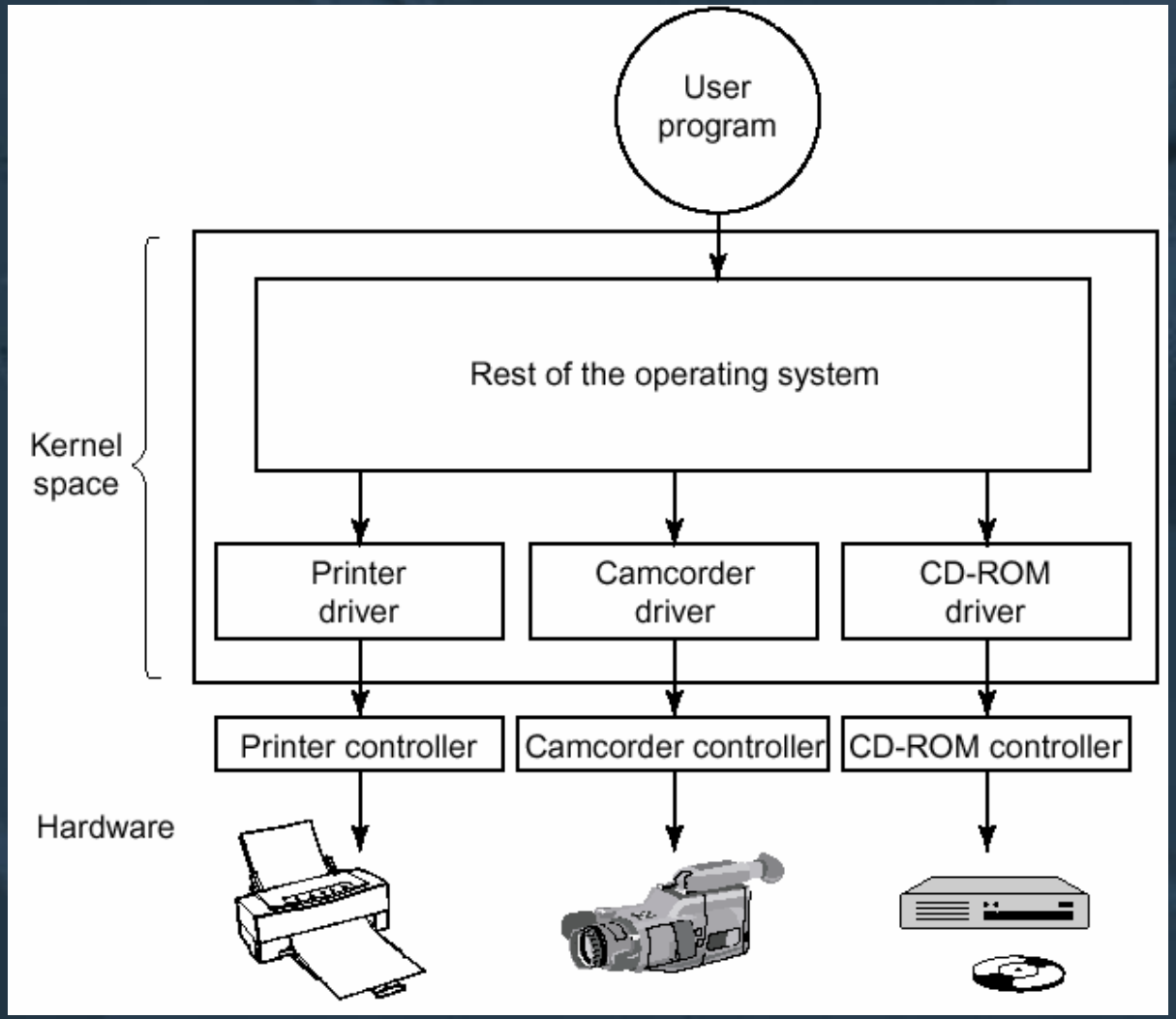
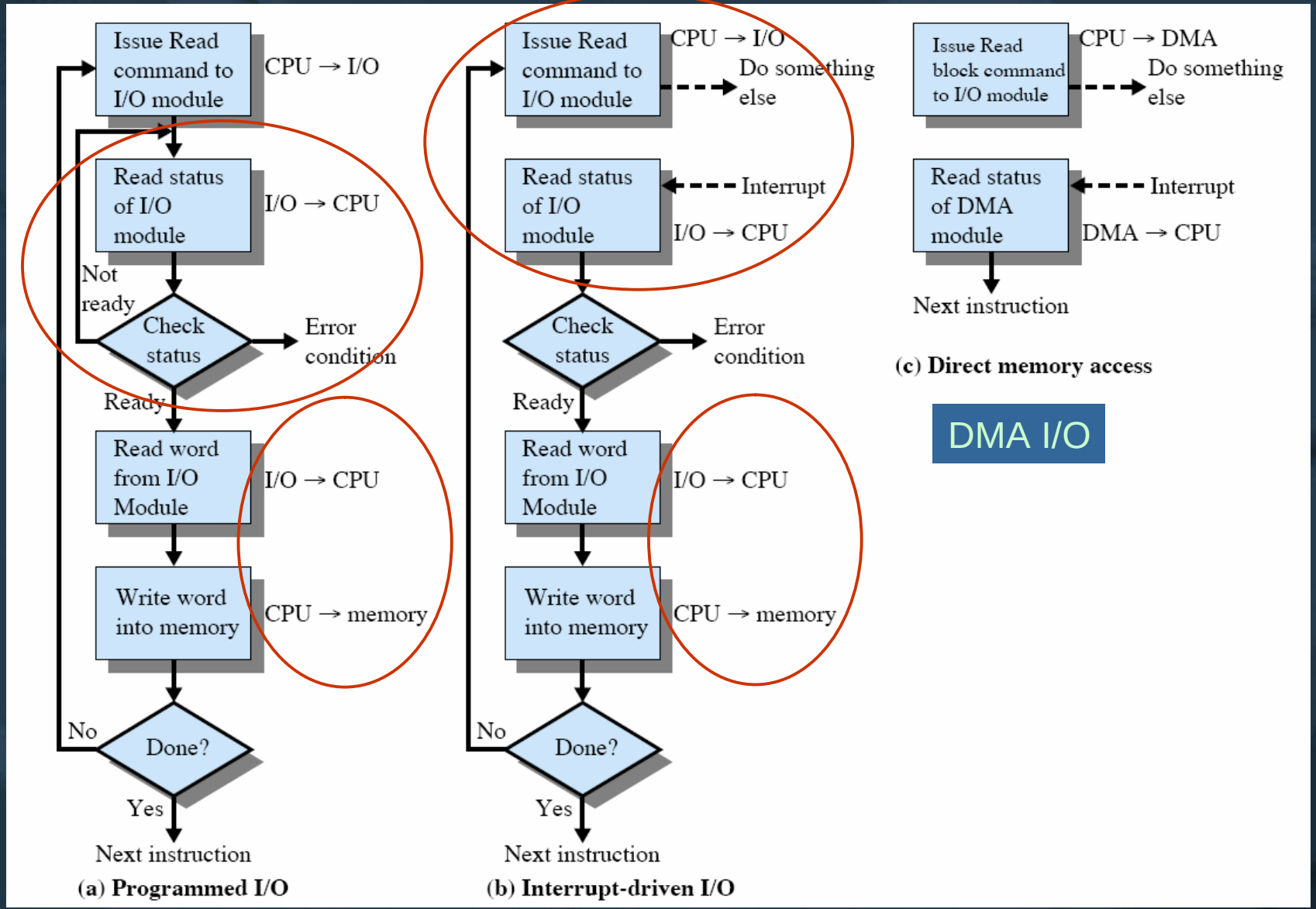


Fig. 5-11 [Tan01]

Set up device registers
check status

Wake up driver
when I/O completed

Kolme eri tapa lukea lohko



DMA I/O

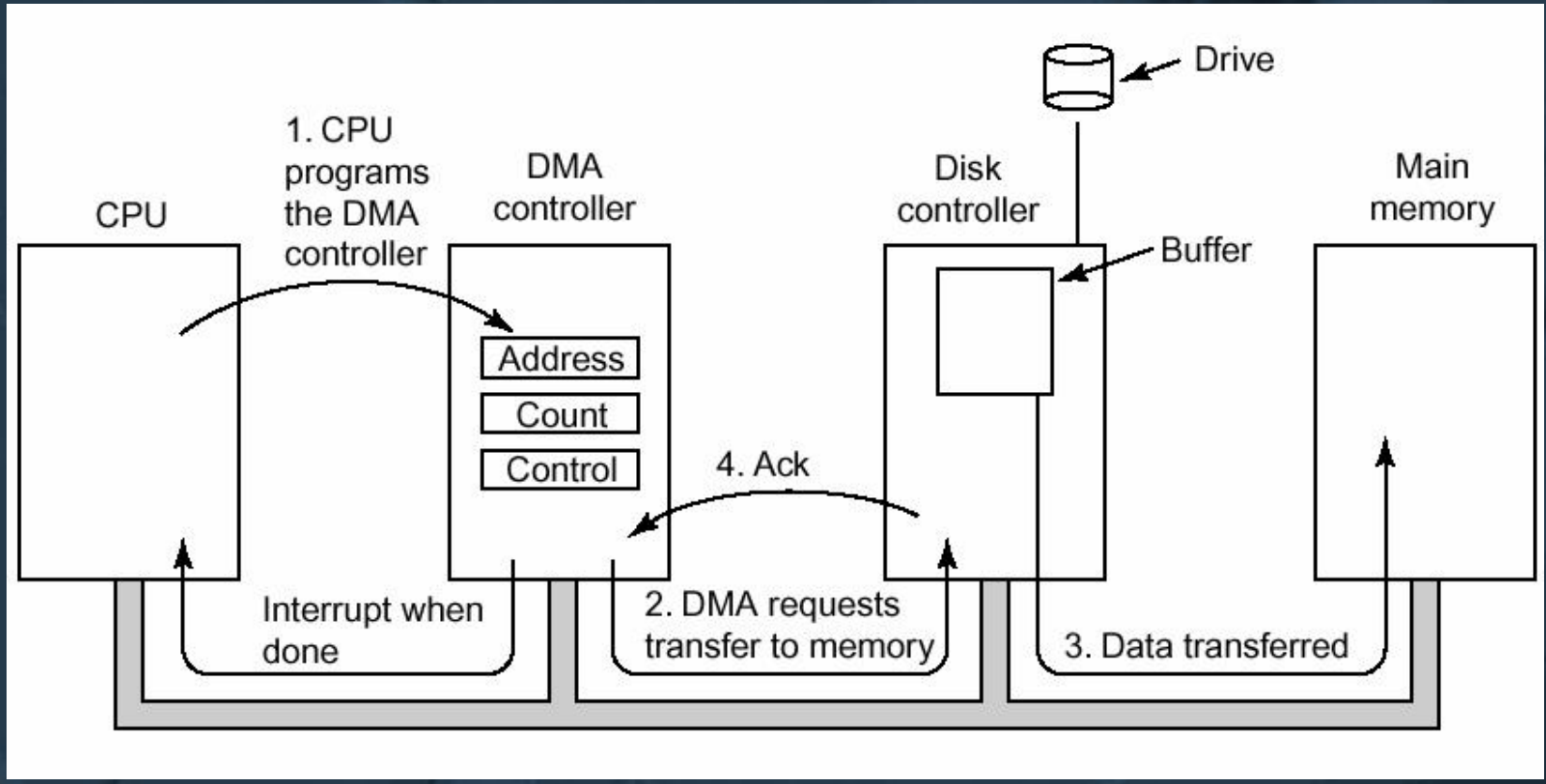
suora I/O

keskeyttävä I/O

Fig 1.19 [Stal05]

DMA-siirto

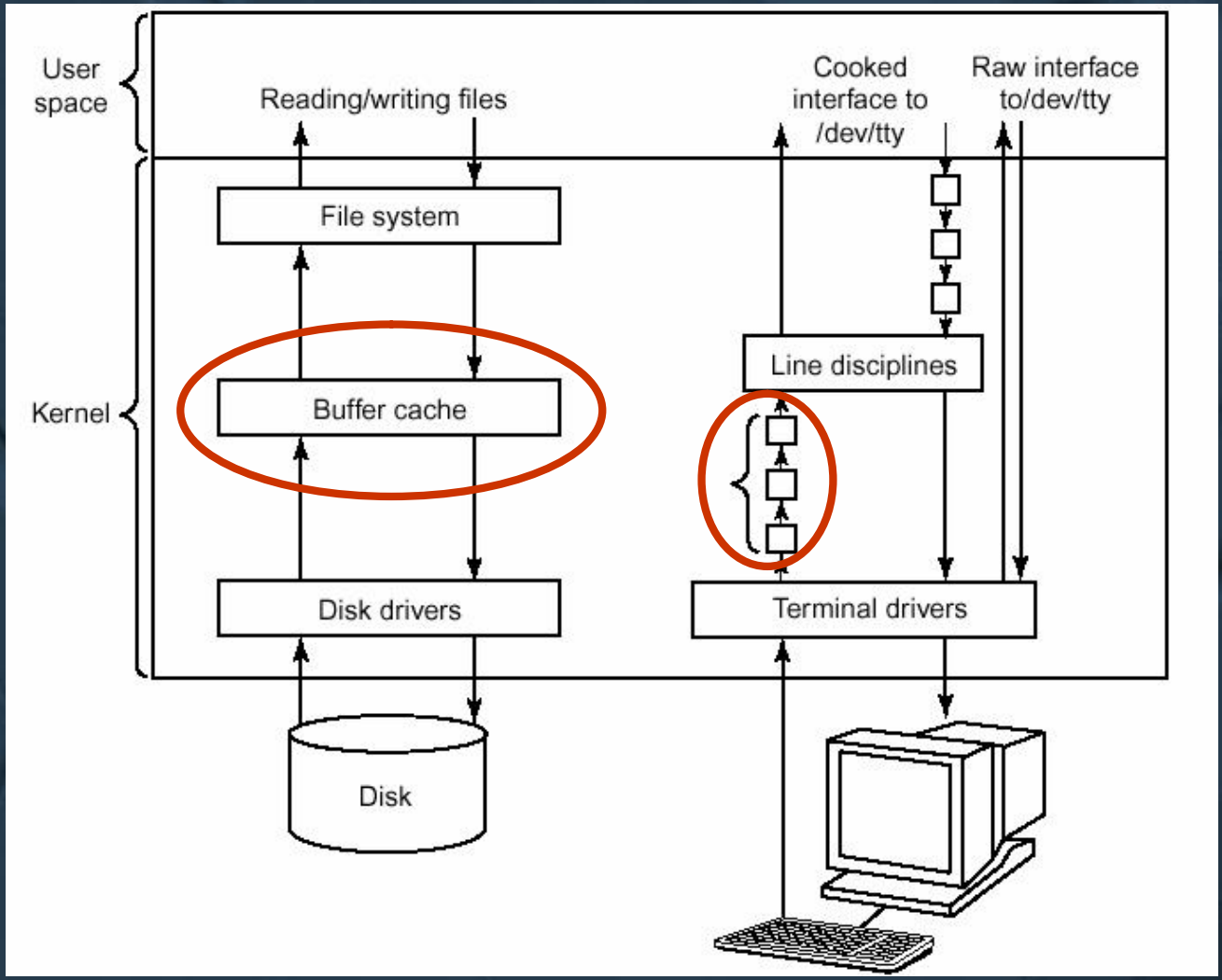
´ Fig. **5-4 [Tan01]**



Huom: data ei kulje suorittimen rekistereiden kautta!

Puskurointi

Fig. 10-22 [Tan01]

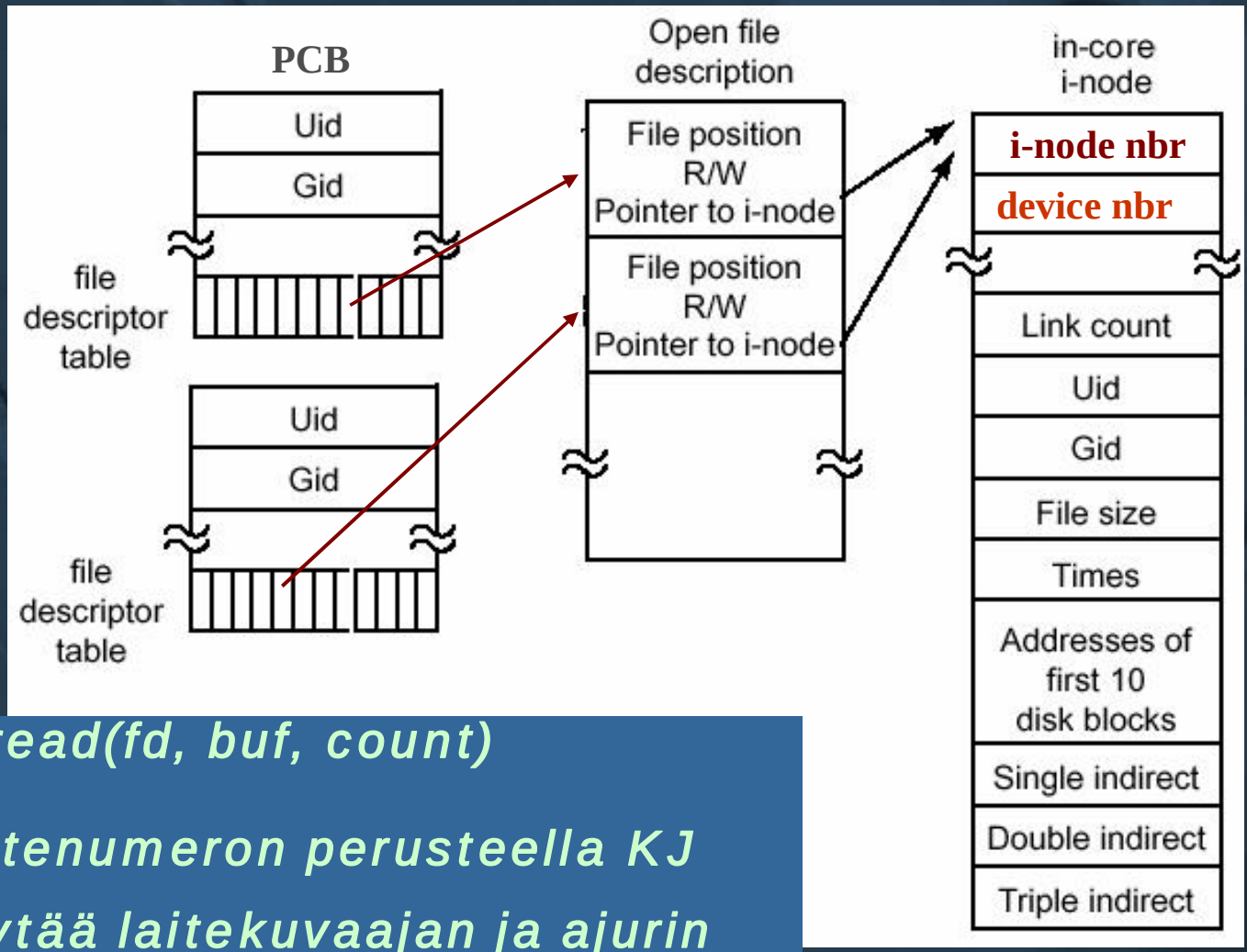


Käyttöjärjestelmät II

Tiedostojärjestelmä

Ch 12.1-6 [Stal05]

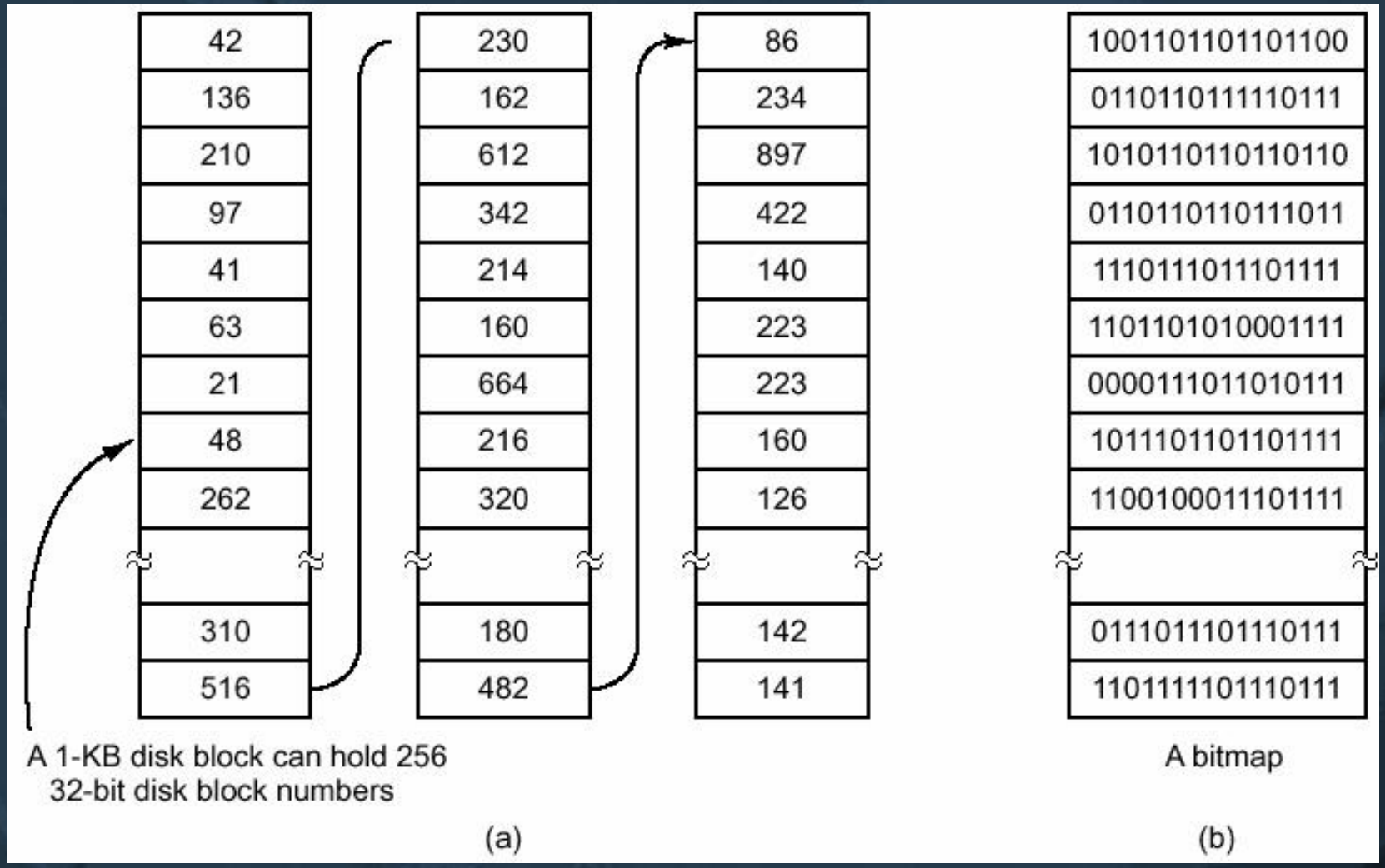
UNIX



n = read(fd, buf, count)

- *laitenumeron perusteella KJ löytää laitekuvaajan ja ajurin*

Vapaan tilan hallinta Fig. 6-21 [Tan01]



linked list vs. bitmap

UNIX

Fig 10-31 [Tan01]

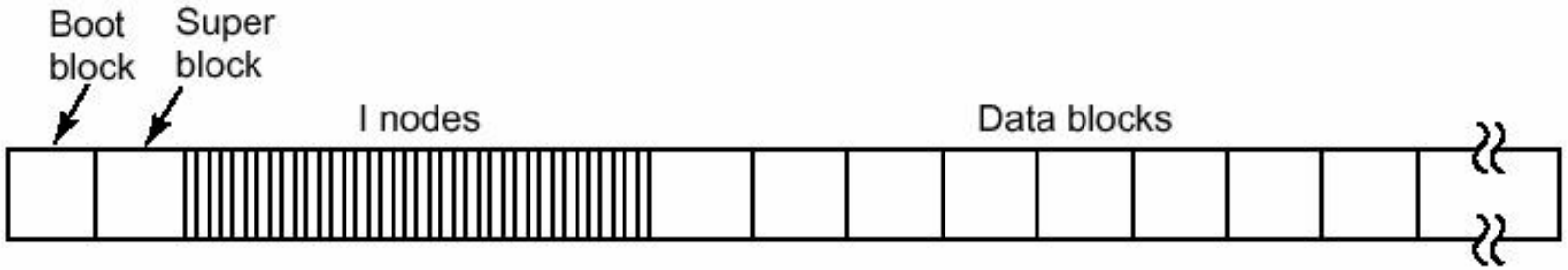
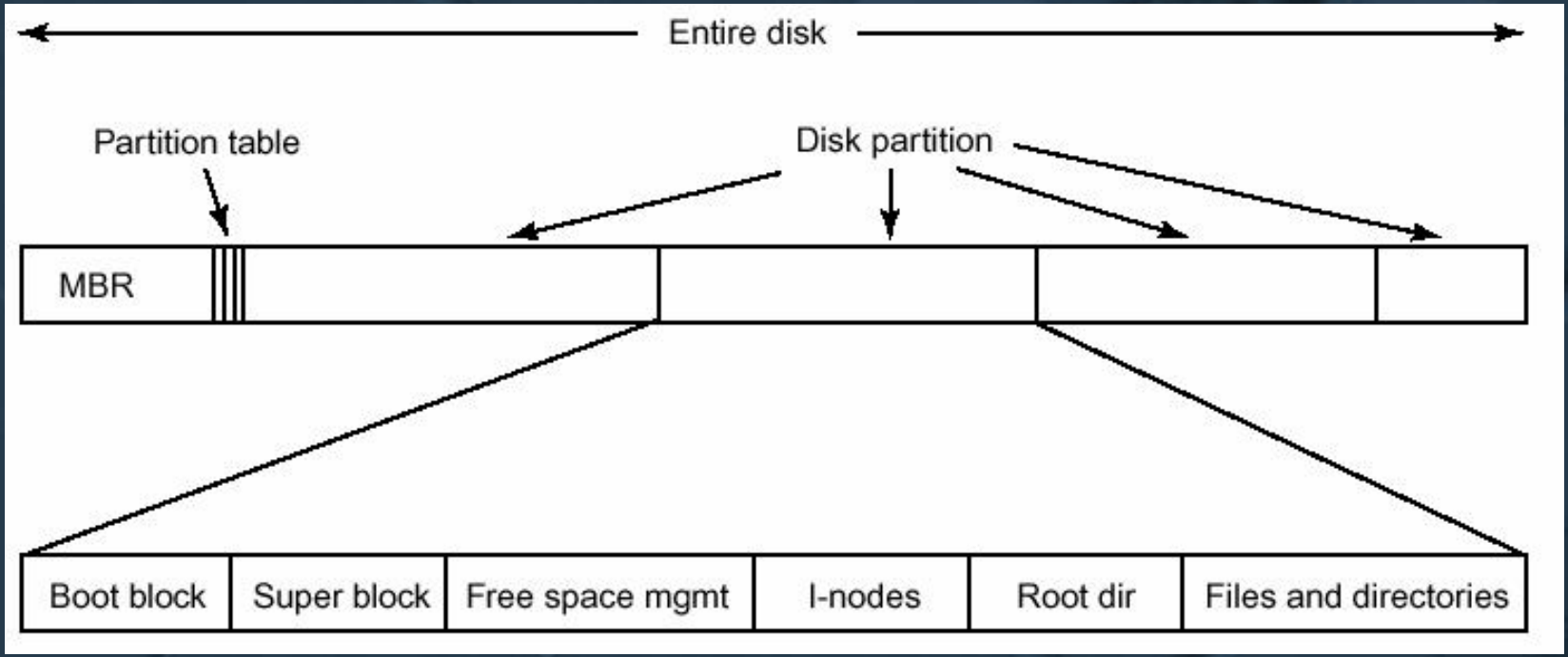


Fig. 10-31. Disk layout in classical UNIX systems.

- vapaat lohkot ketjutettu edellä kuvatulla tavalla taltiota alustettaessa
- i-solmussa merkintä vapaa/varattu
- superlohkossa mm. laitenumero, partition koko sekä vapaiden lohkojen listan alku, vapaiden i-solmujen numeroita

Levypartitiot

Fig. 6-11 [Tan01]



Kertauskysymyksiä

- Mitkä ovat keskeiset KJ:n osat?
- Mitä perustietorakenteita KJ:n on ylläpidettävä?
- Kuinka keskeytys käsitellään?
- Mitä tietoja on prosessin kuvaajassa?
- Milloin noita tietoja käytetään?
- Milloin KJ vaihtaa suoritettavaa prosessia?
- Miten prosessi ja säie liittyvät toisiinsa?
- Mitä hyötyä on asiakas-palvelija mallista?
- Kuinka samanaikaisuutta hallitaan?